REMARKS

INTRODUCTION:

In accordance with the foregoing, claims 1, 5, 6, 7, and 8 have been amended. No new matter is being presented, and approval and entry are respectfully requested.

Claims 1-10 are pending and under consideration. Reconsideration is respectfully requested.

REJECTION UNDER 35 U.S.C. §103:

In the Office Action, at pages 3-9, numbered paragraphs 5-8, claims 1-10 were rejected under 35 U.S.C. §103(a) as being unpatentable over Glass et al. (USPN 6,629128 B1; hereafter, Glass) in view of Howes et al.. (USPN 6,324,277 B1; hereafter, Howes), and further in view of Dugan et al. (USPN 6,425,005 B1; hereafter, Dugan). The reasons for the rejection are set forth in the Office Action and therefore not repeated. The rejection is traversed and reconsideration is requested.

Independent claims 1, 5, 6, 7 and 8 have been amended for clarity.

It is respectfully submitted that Glass teaches a system and method for dynamic generation of remote proxies, which "eliminates the need to parse the .class file, create a java source code file, and shell out the .java file to a compiler since the byte code generation process occurs as part of the dynamic generation of remote proxies" (see Glass, col. 7, lines 27-33 and claim 1). Proxies need to be reflections, or duplicates on the surface, of objects since proxies perform specific tasks such as controlling access to or communications with the objects they represent. Thus, proxies need to look like the object on the outside, but on the inside, proxies contain unique computer code to accomplish their assigned function" (Glass, col. 8, lines 46-52).

Hence, the client system and server system of Glass communicate via a distributed object management system in which a remote proxy class 23 is dynamically generated at the client side to form a client-side representative of the server object, a client-side type generator operates to generate a client-side type object for a class of the server objects, a client-side function generator generates one or more client-side function objects that provide a connection to one or more methods of the server object, a client-side reference generator generates a client-side reference object, a client-side streamer generator generates a set of streamer objects that correspond in number to the methods of the server object wherein one or more of the streamer objects are accessed by the client-side reference object, a server-side reference generator generates a server-side reference object for decoding messages into a format usable by the server-side object broker and to encode messages to transmit to a client-side object

request broker, a server-side local reference generator to generate a local reference object contained within the server-side object broker, a server-side type generator to generator a server-side type of object for the class of the server object wherein the server-side type object is accessed by the server-side reference object using the local reference object, and a server-side function generator to generate one or more server-side function objects, the one or more server-side function objects corresponding in number to the one or more client-side function objects linked to the server-side type object and providing access to methods of the server method (see claim 1, Glass).

Glass requires the use of both a client-side object request broker and a server-side object broker, which combination is not required by the present claimed invention. Independent claims 1, 5, 6, and 7 of the present invention do not recite the use of a client-side object request broker (see, e.g., FIG. 1). Independent claim 8 of the present invention recites an object reference generating device comprising: a connection control unit receiving from an apportioning server: an object reference request having an arrival IP address that is an IP address of the object reference generating unit; and an object reference request that is an apportioning address, wherein the apportioning server has determined that the object reference generating unit is located in a server having a lightest load in comparison with other servers, wherein the connection control unit receives the object reference request for distribution of a naming service in CORBA initially sent from a client; an interface apportioning unit receiving connection information from the connection control unit and apportioning an interface within an Object Request Broker (ORB); a naming service unit to dynamically generate a naming service object reference with address information corresponding to request time connection information; and the ORB performing interface processing between the interface apportioning unit and the naming service unit to distribute a load by allocating an IP address that applies a naming service to load distribution using an apportioning server.

It is respectfully submitted that Glass's use of a remote proxy system is different from the present invention's dynamic generation of a naming service and dynamic setting of address information into the object reference. Hence, Glass teaches away from the present invention.

Howes does not teach or suggest dynamic generation of a naming service and dynamic setting of address information into the object reference wherein load balancing is taking place, as is disclosed in amended independent claims 1, 5, 6, 7, and 8 of the present invention. Howes instead teaches a method and apparatus for managing connections based on client IP address (see Abstract, Howes). Hence, Howes teaches away from the present invention, and it would not have been obvious to combine Howes with Glass.

Dugan teaches a method and apparatus for managing local resources and service nodes in an intelligent network, not dynamic generation of a naming service and dynamic setting of

address information into the object reference wherein load balancing is taking place, as is disclosed in amended independent claims 1, 5, 6, 7, and 8 of the present invention. Dugan utilizes a first level processor for providing one or more local execution environments at a node, each object execution environment including a mechanism for instantiating and executing one or more service objects capable of performing services related to a received event communicated to said node, said first level processor further generating status information relating to each said local execution environment; and a second level processor associated with a service node and communicably linked to said first level processor for receiving said status information and tracking a processing capability of each local execution environment, said second level processor determining which instance of a requested service is to be implemented at said first level based upon said capability and availability of service objects (see claim 1, Dugan). However, Dugan does not teach or suggest the present invention's dynamic generation of a naming service and dynamic setting of address information into the object reference. Hence, Dugan teaches away from the present invention. Thus, there is no teaching or suggestion of combining Glass, Dugan and Howes.

Because of their different implementations, it is respectfully submitted that it would not be obvious to combine Glass, Howes and Dugan, which do not fit together to teach or suggest the present invention's dynamic generation of a naming service and dynamic setting of address information into the object reference.

Hence, it is respectfully submitted that amended claims 1, 5, 6, 7 and 8 are patentable under 35 U.S.C. §103(a) over Glass et al. (USPN 6,629128 B1) in view of Howes et al. (USPN 6,324,277 B1), and further in view of Dugan et al. (USPN 6,425,005 B1). Since claims 1, 3, 4, 9 and 10 depend from amended claims 1 and 5, respectively, claims 1, 3, 4, 9 and 10 are submitted to be patentable under 35 U.S.C. §103(a) over Glass et al. (USPN 6,629128 B1) in view of Howes et al. (USPN 6,324,277 B1), and further in view of Dugan et al. (USPN 6,425,005 B1) for at least the reasons that amended claims 1 and 5 are patentable under 35 U.S.C. §103(a) over Glass et al. (USPN 6,629128 B1) in view of Howes et al. (USPN 6,324,277 B1), and further in view of Dugan et al. (USPN 6,425,005 B1).

CONCLUSION:

In accordance with the foregoing, it is respectfully submitted that all outstanding objections and rejections have been overcome and/or rendered moot, and further, that all pending claims patentably distinguish over the prior art. Thus, there being no further outstanding objections or rejections, the application is submitted as being in condition for allowance which action is earnestly solicited.

If the Examiner has any remaining issues to be addressed, it is believed that prosecution can be expedited by the Examiner contacting the undersigned attorney for a telephone interview to discuss resolution of such issues.

If there are any underpayments or overpayments of fees associated with the filing of this Amendment, please charge and/or credit the same to our Deposit Account No. 19-3935.

Respectfully submitted,

STAAS & HALSEY LLP

Date: December 21, 2006 By: Darleen J. Stockley

Darleen J. Stockley
Registration No. 34,257

1201 New York Avenue, N.W.
Suite 700
Washington, D.C. 20005
Telephone: (202) 434-1500
Facsimile: (202) 434-1501